UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/682,133	10/09/2003	Richard D. Dettinger	ROC920030237US1	1361

46797 7590 03/30/2010

IBM CORPORATION, INTELLECTUAL PROPERTY LAW
DEPT 917, BLDG. 006-1
3605 HIGHWAY 52 NORTH
ROCHESTER, MN 55901-7829

EXAMINER	
PHAM, KHANH B	

ART UNIT	PAPER NUMBER
2166	

MAIL DATE	DELIVERY MODE
03/30/2010	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte RICHARD D. DETTINGER and RICHARD J. STEVENS

Appeal 2009-005164
Application 10/682,133
Technology Center 2100

Decided: March 30, 2010

Before JAMES D. THOMAS, LEE E. BARRETT, and HOWARD B. BLANKENSHIP, *Administrative Patent Judges*.

BLANKENSHIP, *Administrative Patent Judge*.

DECISION ON APPEAL

STATEMENT OF THE CASE

This is an appeal under 35 U.S.C. § 134(a) from the Examiner's final rejection of claims 1-17, 26-41, 50, and 51, which are all the claims remaining in the application. We have jurisdiction under 35 U.S.C. § 6(b).

We affirm-in-part.

Invention

Appellants' invention relates to managing composite query operations. A composite query operation combines two or more simple query operations; e.g., the simple query operations of UPDATE and INSERT are combined in an UPSERT operation. If an UPDATE operation fails because a record is not found, an INSERT operation is performed. Spec ¶¶ [0005], [0008]. Appellants describe logic for determining what part of composite query operations tend to fail, such that appropriate query operations can be selected to avoid needless query repetition. *Id.* at ¶¶ [0034], [0071]-[0073].

Representative Claims

1. A method of managing execution of query operations in a data processing system, comprising:

issuing, by a requesting entity, a request to perform a composite query operation defined by at least an initial query operation and a plurality of subsequent query operations to be executed against a data repository of the data processing system;

executing the initial query operation;

determining an operation status of the initial query operation;

selecting one of the plurality of subsequent query operations based on the operation status;

performing the selected subsequent query operation;

updating the operation status based on a result of the subsequent query operation;

managing execution of any remaining subsequent query operations on the basis of the updated operation status; and

upon determining the composite query operation has completed, returning a result of the composite query operation to the requesting entity.

9. A method of managing execution of query operations in a data processing system, comprising:

issuing, by a requesting entity, a request to perform a composite query operation defined by at least an initial query operation and a plurality of subsequent query operations to be executed against a data repository of the data processing system;

providing selection logic defining a next query operation of the composite query operation to be executed;

providing a plurality of failure conditions for determining when a failure of the composite query operation occurs;

managing, using a composite query operations manager, execution of the initial query operation and the plurality of subsequent query operations on the basis of the selection logic and the plurality of failure conditions; and

upon determining the composite query operation has completed, returning a result of the composite query operation to the requesting entity.

Prior Art

Scalzo, *Oracle DBA Guide to Data Warehousing and Star Schema*, Prentice Hall, June 4, 2003, p. 1-10, available at http://proquest.safaribooksonline.com/0130325848

Examiner's Rejections

Claims 1-17, 26-41, 50, and 51 stand rejected under 35 U.S.C. § 102(a) as being anticipated by Scalzo.

Claim Groupings

In accordance with Appellants' arguments and claim groupings in the Appeal Brief, we will decide the appeal on the basis of claims 1 and 9. Claim 1 will represent claims 1-8, 26-32, and 50. Claim 9 will represent claims 9-17, 33-41, and 51. *See* 37 C.F.R. § 41.37(c)(1)(vii).

FINDINGS OF FACT

In "Scenario 4," Scalzo describes a composite query ("upsert") in which a record may be updated, but a record is inserted when an existing record is not found for an update. Scalzo at 6-8. In particular, Scalzo discloses program code for preparing and executing "upsert." *Id.* at 7-8.

PRINCIPLES OF LAW

The *claims* measure the invention. *See SRI Int'l v. Matsushita Elec. Corp.*, 775 F.2d 1107, 1121 (Fed. Cir. 1985) (en banc). Our reviewing court has repeatedly warned against confining the claims to specific embodiments described in the specification. *Phillips v. AWH Corp.*, 415 F.3d 1303, 1323 (Fed. Cir. 2005) (en banc). During prosecution before the USPTO, claims are to be given their broadest reasonable interpretation, and the scope of a claim cannot be narrowed by reading disclosed limitations into the claim. *See In re Morris*, 127 F.3d 1048, 1054 (Fed. Cir. 1997); *In re Zletz*, 893 F.2d 319, 321 (Fed. Cir. 1989); *In re Prater*, 415 F.2d 1393, 1404-05 (CCPA 1969). "An essential purpose of patent examination is to fashion claims that are precise, clear, correct, and unambiguous. Only in this way can uncertainties of claim scope be removed, as much as possible, during the administrative process." *In re Zletz*, 893 F.2d at 322.

"Anticipation requires the presence in a single prior art reference disclosure of each and every element of the claimed invention, arranged as in the claim." *Lindemann Maschinenfabrik GMBH v. American Hoist & Derrick Co.*, 730 F.2d 1452, 1458 (Fed. Cir. 1984).

ANALYSIS

Claim 1

The Examiner finds that Scalzo's executing the "fgets" command to update operation status corresponds to updating the operation status based on a result of the subsequent query operation as recited in claim 1. Ans. 4. The Examiner explains further that after each iteration of the "while" loop, the "fget" is executed and returns the next record in the inputted file. The return record is compared with "NULL" to determine current operation status. At the end of the inputted, file, "fget" will return NULL and the operation status is changed to "FALSE" to indicate that the last record has been read and the program will exit the "while" loop. According to the Examiner, Scalzo updates the operation status from "TRUE" to "FALSE" to cause the program exit from the loop. Otherwise, if the operation status were not changed, the "while" loop would run indefinitely. *Id.* at 10-11.

However, claim 1 recites updating the operation status "based on a result of" the subsequent query operation. In the Examiner's scenario, the operation status in Scalzo is updated based on the last record being read. The query operation result does not affect the update status; the update status changes regardless of what the query operation result may be. We therefore conclude that execution of "fgets" does not meet the requirement of updating the operation status as claimed.

We thus agree with Appellants that the rejection fails to demonstrate anticipation of claim 1 by Scalzo. Claims 2-8 depend from claim 1. Each of independent claims 26 and 50 contains the same limitation of claim 1 that we have considered but is rejected on the same basis (Ans. 8). Claims 27-32 depend from claim 26. We therefore do not sustain the § 102(a) rejection of claims 1-8, 26-32, and 50.

Claim 9

Appellants submit that Scalzo does not describe "managing, using a composite query operations manager, execution of the initial query operation and the plurality of subsequent query operations on the basis of the selection logic and the plurality of failure conditions." App. Br. 14-15. Although the Examiner discusses the code disclosed at pages 7 and 8 of the reference, Appellants submit that Scalzo in no way discloses that "each pass" through the while loop is performed on the basis of the selection logic and the plurality of failure conditions. According to Appellants, the Scalzo code does not show a "plurality of failure conditions." *Id.* at 15.

Appellants acknowledge (e.g., Reply Br. 2-3) that the Scalzo code performs an "upsert" operation. The "upsert" operation in Scalzo is consistent with the "upsert" operation described in Appellants' Specification. That is, the logic is designed such that if an INSERT query operation fails, an UPDATE query operation is executed. *See* Spec. ¶¶ [0071]-[0072]. Appellants' remarks in the briefs, although nominally in support of claim 9, seem to argue against the "fgets" command as applied against claim 1. Appellants' arguments appear not to respond to the

Examiner's basis for finding that the Scalzo's code describes logic that meets the requirements of the "managing" step of claim 9.

Scalzo discloses the following "while" loop that performs the "upsert" operation at pages 7 and 8.

```
/* Process data file records */
while (fgets (rec, sizeof rec, fid) != NULL) {
 . . .
  /* Obtain adjustment via lookup */
EXEC select  adj_value
        into     :h_av:i_av
        from    lookup_table
        where   adj_lookup = :h_c2:i_c2;
 if (sqlca. sqlcode == 1403) {
   h_av = 0;
   i_av = 0;
 }
 else if (sqlca.sqlcode != 0) {
  . . .
 }

/* First - try to update existing record */
EXEC SQL EXECUTE update_ command
      USING :h_c1:i_cl,
                :h_c2:i_c2,
                :h_c3:i_c3,
                :h_c4:i_c4,
                :h_c5:i_c5,
                :h_av:i_av;

/* Second - if update fails because record
                not found, then insert record*/
if (sqlca.sqlcode == 1403) {
  EXEC SQL EXECUTE insert_command
      USING :h-cl:i-cl,
                :h_c2:i_c2,
                :h_c3:i_c3,
                :h_c4:i_c4,
```

```
                    :h_c5:i_c5,
                    :h_av:i_av;
      }
      else if  (sqlca.sqlcode != 0 {
       . . .
      }
       . . .
    }
     . . .
}
```

Scalzo code at 7-8 (emphasis added).

As shown above, there are two occurrences of the determination "if (sqlca.sqlcode == 1403)." As noted in the code comments, the second occurrence tests whether a record update has failed, such that a record is to be inserted; i.e., "/* Second - if update fails because record not found, then insert record*/." The code thus tests for a "failure condition" in accordance with Appellants' claim 9. In Scalzo, there is a failure in the composite query (update or insert) when the update fails.

In Scalzo's code, the "sqlca.sqlcode" is also tested prior to execution of the "upsert" operation. Both occurrences are inside the "while" loop, and are thus relevant in "each pass" through the "while" loop. Moreover, the first occurrence also relates to a "failure condition," because the test is identical to that of the subsequent test.

Contrary to Appellants' arguments, Scalzo thus describes a plurality, or at least two, failure conditions such that "each pass" through the while loop is performed on the basis of the selection logic and the plurality of failure conditions. Appellants have failed to show that claim 9 has been rejected in error. Further, as the Examiner indicates, the broadly drafted "execution of the initial query operation and the plurality of subsequent

query operations" does not distinguish over the initial query operation with respect to a first record in a file and a plurality of subsequent query operations with respect to following records in the file. The law of anticipation does not require that a reference "teach" what an applicant's disclosure teaches. Assuming that a reference is properly "prior art," it is only necessary that the claims "read on" something disclosed in the reference, i.e., all limitations of the claim are found in the reference, or are "fully met" by it. *Kalman v. Kimberly-Clark Corp.*, 713 F.2d 760, 772 (Fed. Cir. 1983).

We therefore sustain the rejection of claim 9. Claims 10-17, 33-41, and 51 fall with claim 9. *See* 37 C.F.R. § 41.37(c)(1)(vii).

DECISION

The rejection of claims 1-17, 26-41, 50, and 51 under 35 U.S.C. § 102(a) as being anticipated by Scalzo is reversed with respect to claims 1-8, 26-32, and 50 but affirmed with respect to claims 9-17, 33-41, and 51.

No time period for taking any subsequent action in connection with this appeal may be extended under 37 C.F.R. § 1.136(a). *See* 37 C.F.R. § 41.50(f).

AFFIRMED-IN-PART

rwk

IBM CORPORATION, INTELLECTUAL PROPERTY LAW
DEPT 917, BLDG. 006-1
3605 HIGHWAY 52 NORTH
ROCHESTER, MN 55901-7829